

HOW IT WORKS

DirectX 10

Without DirectX, games would have to be written specifically for your hardware. Simon Handby explains how DirectX 10 will take the technology even further

It's not a thought that generally occurs when you're spraying aliens with gunfire or trying to out-brake another driver into Monza's famous Parabolica corner, but computers have evolved into a very impressive gaming platform. It's not just gaming, either. More of us are using our PCs to listen to music, watch films or even catch up with the week's TV, to the extent that traditional media companies are struggling to adjust to our demands. Faster, always-on internet access has helped to drive the changes, along with ever more powerful hardware. One can forget how much easier it's become to get fancy multimedia applications working in the first place.

Under DOS, a computer required a driver for each bit of non-standard hardware, such as a sound card, which at the time was a luxury. Programs written for the PC could access such devices directly, but only if the developers knew how to give instructions to a specific driver. Consequently, not all games supported all hardware. Moreover, it was sometimes necessary to spend hours tinkering with the Config.sys and Autoexec.bat files to free up memory for the extra drivers, as gamers who had DOS are likely to remember. Things weren't much better with the early versions of Windows, which ran on top of a DOS shell. Most developers continued to write games that ran in a DOS session.

From Windows 95 onwards, though, the operating system no longer sat on top of a DOS session, and its protected memory mode meant developers no longer had direct access to hardware. Microsoft realised that for Windows to be an effective gaming and multimedia platform, programmers would need a way to send advanced instructions to multimedia hardware without having to do so through the operating system's comparatively slow high-level routines. The first version of DirectX, released in September 1995, was Microsoft's solution to the problem.

FUNCTION ROOM

DirectX comes as a single package to install, but it's actually a collection of several application programming interfaces (APIs), each of which provides access to a different multimedia function. These include DirectSound for sound, DirectInput for game controllers, DirectDraw for 2D graphics and Direct3D for 3D graphics.

As we've already mentioned, a programmer developing applications for DOS would have directly accessed the driver for each multimedia device that they wanted to use. Not only is this prevented in modern versions of Windows, but it would be impossible to learn how to 'speak' to the full range of products available today. One of DirectX's most important roles is to act as a translator between applications and the devices that they need (see The Role of Drivers, right).

Clearly, DirectX adds an extra stage to the process, but it's a powerful one. DirectX-compatible hardware comes with a driver that conforms to a standard laid down by Microsoft, so instructions and data can be freely exchanged between the device and the API. Similarly, developers need only learn how to pass instructions to DirectX, which translates them into a language that the hardware understands, solving many compatibility problems at a stroke.

However, while an expensive graphics card might race through any task that a programmer cares to throw at it, a cheaper device may not support certain features such as advanced lighting or texturing (clothing an object with an image). Directly asking a device to do the impossible would inevitably cause its driver or the application to crash.

DirectX queries the capabilities of the hardware that it finds, and is able to manage if a device doesn't support a function that a programmer has called for (see Hardware Abstraction Layer, opposite). If a routine isn't available in hardware, DirectX emulates the necessary step in software so that the requesting application doesn't grind to a halt. Known as hardware abstraction, and performed by the hardware abstraction layer (HAL), this step relies on the system processor. As this doesn't have the same hardware optimisations as a dedicated graphics or sound chip, the results may look or sound more basic.

Although hardware acceleration is still the ideal, the HAL means that programmers don't have to concern themselves with the exact specifications of a system's hardware. However, it's prudent to keep requests largely within a device's hardware capabilities, so most games have a choice of detail levels and may offer a recommended setting.

VERSION TERRITORY

Many people groan at the thought of having to upgrade software, but multimedia hardware evolves quickly, with new features and abilities being developed regularly. As a result, DirectX has undergone many revisions in the 12 years since it was introduced.

Each successive version has been backwards-compatible with earlier hardware, so it shouldn't be a problem if, for example, you have a sound card from 2001 but a brand-new graphics card. In addition, each new version of DirectX has to date been backwards-compatible with applications written for earlier releases. Therefore, a game written for DirectX 8.1, which launched with Windows XP in October 2001, is likely to run successfully on a computer with Windows Vista and DirectX 10.

It's tempting to imagine Microsoft's coders frantically trying to roll out new versions of

DirectX to keep pace with hardware innovations, but new releases aren't quite so straightforward. Because DirectX provides a set of instructions that developers and hardware manufacturers need to support, it serves as a *de facto* standard that defines, and sets limits on, what can be achieved. It might become possible to build hardware that supports an exciting new feature, but there isn't really much point in introducing it until DirectX has been updated to allow its use.

Accordingly, most versions of DirectX have had a short life of around a year. That said, DirectX 9, which was introduced at the end of 2002, remains the latest version for users of Windows 98, Me and XP nearly five years on. Although graphics processors have become hugely more powerful in this time, with quicker clock speeds and far higher memory bandwidth, the most recent DirectX 9-compatible cards have few features that are not found in the first examples.

Part of the reason for the long life of DirectX 9 is that, during its lifetime, Microsoft moved away from its Common Object Model (COM) object-oriented programming technology, adopting the more simple .NET development framework. DirectX 9.0b was the most current version when we last looked at DirectX in *How It Works*, *Shopper* September 2004. We described the associated reworking of the library as "a brief pause in the progress of DirectX". Indeed, DirectX 9.0c was released as that issue went on sale, but by this time Microsoft was already well advanced with the development of Windows Vista.

TOP TEN

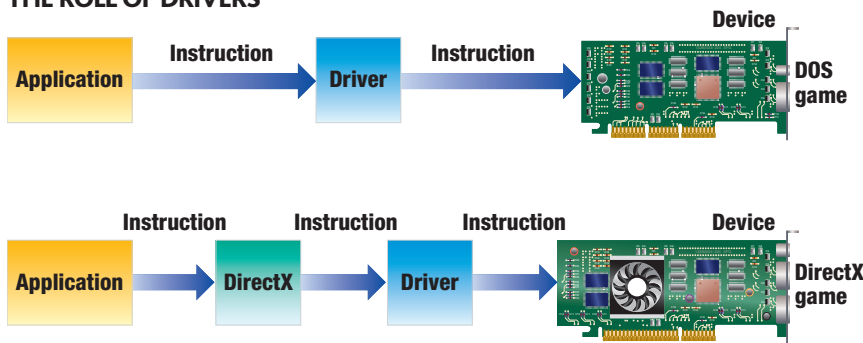
During the development of Vista, Microsoft chose to substantially redesign the way video, audio and other multimedia features work within Windows, which now uses a different driver model to previous versions. DirectX 10, which launched with the new operating system at the end of 2006, is very different from DirectX 9, and forms an essential part of the multimedia changes that Microsoft implemented.

Vista's so-called Windows Display Driver Model (WDDM) enables the desktop's advanced graphical features and, Microsoft says, provides greater stability and fault-tolerance. One of the biggest graphical changes the system introduced is virtualisation, which allows a single graphics card to appear as different logical devices to multiple applications.

Windows XP users who were looking forward to upgrading to the latest graphics card and playing DirectX 10 games have been disappointed. Although Microsoft announced in September that it was extending sales of the operating system until June 2008, DirectX 9 will remain the last compatible release for Windows

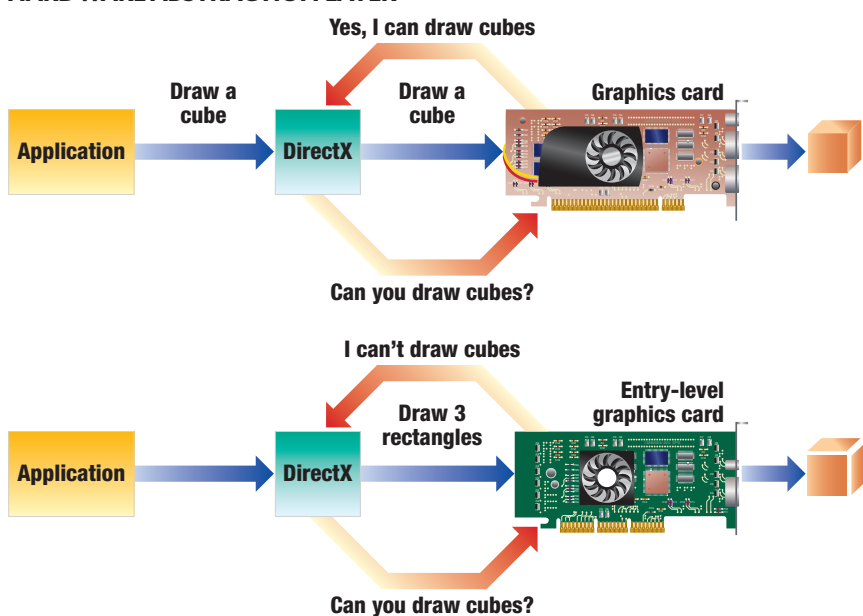
Direct approach How DirectX works in practice

THE ROLE OF DRIVERS



▲ With DOS games, the application would have to talk directly to the graphics card's driver in order to control the hardware (top). With Windows, DirectX sits between the application and the driver, presenting a common interface (bottom). This way, games need only to be written to use DirectX, not specific hardware.

HARDWARE ABSTRACTION LAYER



▲ The hardware abstraction layer (HAL) means that sophisticated graphics can be displayed using cards of varying capabilities. Here we want to draw a cube. With a more capable card (top), the instruction can be passed directly to the hardware. With a simpler card (bottom), the HAL converts the cube command into several simpler instructions that can be handled by the hardware.

XP. An FAQ answer on the Microsoft developer network (MSDN) site states: "Since [the new versions of DirectX] rely on the WDDM technology, they will never be available on earlier versions of Windows."

Arguably the most controversial change in DirectX 10 is the way in which sound is handled. The components that provided accelerated audio functions in earlier Windows releases can no longer access hardware acceleration under Vista. DirectSound, which manages the production of sounds and music, now runs in software emulation. DirectSound3D, which enabled complex environmental effects such as echoing and the 3D placement of audio, is no longer supported at all.

These major changes mean that older applications that use DirectX audio functions, such as games and DVD playback software, will present the processor with extra work under Windows Vista. However, by removing the hardware acceleration, Microsoft is aiming to encourage developers of new applications to use the alternative XACT library. This succeeds

DirectSound and includes X3DAudio for environmental effects.

NEW FEATURES

While the audio overhaul may have proved tricky for some users, other changes in DirectX 10 have been more welcome. One area of improvement is the way in which video hardware handles shaders, which are accelerated routines that apply lighting, shadows and textures to the geometrical components of a scene. Where previous versions of DirectX and non-Microsoft 3D libraries such as OpenGL have used separate pixel (point) and vertex (corner) shaders, DirectX 10 requires graphics cards to have a unified shader architecture. These cards, such as Nvidia's GeForce8 series, have a single set of shaders that can be used flexibly according to the requirements of the workload. This can improve performance dramatically.

For example, a DirectX 10 graphics card can use most of its shaders as pixel shaders when drawing a scene inside, as there are fewer objects to draw and more detail to take care of; outside,

it can set most of its shaders as vertex shaders, as there are more objects to draw. The net result is fast rendering no matter what the scene. With older DirectX hardware, the fixed number of inflexible shaders meant that performance depended on the scene.

Another advantage of what Microsoft calls Shader Model 4.0 is that it supports longer, more sophisticated instructions. This means the graphics processor can be used for complex physics calculations that enable in-game objects to move more realistically, without taxing the processor. Microsoft says DirectX 10 delivers "up to five times" the graphical performance of DirectX 9, though the consensus on gaming forums appears to be that the developers of games and hardware drivers have some catching up to do before that potential is realised. For compatibility with older applications, Vista also ships with modified versions of DirectX 9.

BOX SOLID

DirectX became so successful a platform for PC games that DirectX 8.1 was chosen as the basis for Microsoft's Xbox games console, released in the US at the end of 2001. Indeed, the console was originally codenamed DirectXbox. With standardised hardware very similar to a contemporary PC, the Xbox used a simplified and non-upgradable version of DirectX. The resulting architecture and APIs were familiar to PC programmers, and the Xbox was considered an easy console to develop for.

Whereas the original Xbox used a Pentium III processor, the Xbox 360's hardware is unique to the console and quite different from a PC's. However, several modifications to the way that applications control hardware on the Xbox 360 have made their way into DirectX 10, as Microsoft attempts to provide developers with an integrated development platform that can be used to produce games for both the Xbox 360 and the PC.

In December last year, Microsoft released the downloadable XNA framework, which aims to make Xbox 360 development more accessible to hobbyists and smaller companies. The XNA Game Studio Express development software is available to download for the PC, and Xbox 360 owners can also subscribe to the XNA Creators Club via the Xbox Live Marketplace. Although this doesn't provide a full professional games development environment, it lets individuals use their console to create, debug and play games for a fraction of the price of a professional development kit. ☐

FURTHER INFORMATION

Wikipedia entry on DirectX http://en.wikipedia.org/wiki/Direct_x

Computer Shopper interview with a games programmer www.computershopper.co.uk/news/127469

Microsoft blog comparing DirectX 10 with DirectX 9 www.tinyurl.com/ymcgch

DirectX page on Microsoft Developer Network <http://tinyurl.com/2n2cce>

Huge amount of detail on Xbox 360 <http://tinyurl.com/2gfo8w>

Microsoft's Coding4Fun resource for those who fancy trying their hand at programming <http://tinyurl.com/34xk9t>